REMARKS

Claims 2, 3, 4, 5, 7, 8, 9, 10, 11, 13, 14, and 15 were pending in the application at the time of examination. Claims 2, 3, 4, 5, 7, 8, 9, 10, 11, 13, 14, and 15 stand rejected as anticipated.

1.0 REJECTION OF CLAIMS 2, 3, 4, 5, 7, 8, 9, 10, 11, 13, 14, and 15 UNDER 35 U.S.C. 102(b)

The Examiner rejected Claims 2, 3, 4, 5, 7, 8, 9, 10, 11, 13, 14, and 15 as anticipated by U.S. Patent No. 3,701,111 to Cocke et al., (hereinafter, "Cocke").

Applicant respectfully traverses the anticipation rejection of Claim 2.

Applicant respectfully asserts that the Office Action failed to show that Cocke teaches, discloses, or suggests the invention as recited in Claim 2.

To anticipate a claim, the MPEP directs:

> **TO ANTICIPATE A CLAIM, THE REFERENCE MUST TEACH EVERY ELEMENT OF THE CLAIM**
>
> A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference. ... The identical invention must be shown in as complete detail as is contained in the ... claim.

MPEP, §2131, 8th Ed., Rev. 4, p. 2100-76 (October 2005).

Claim 2 recites in part:

> ...providing a unique variable bit length binary representation of the absolute value of said integer data;

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

> appending to said unique variable bit length binary representation a single bit representing the sign of said integer data;
> wherein said single bit is zero for integer data that is less than or equal to zero.

The Office Action indicated that Cocke discloses a technique for decoding and encoding digits into bits using a sign bit to show a negative or positive number, and that Figures 4 and 10 clearly show the idea.

Cocke describes Figure 4 as:

> FIG. 4 is a diagrammatic representation of a decoding scheme **showing the manner in which the primary and secondary processors P and S are employed to decode individual variable-length codes independently of one another**, this mode of operation being referred to herein as "independent decoding" to distinguish it from "dependent decoding" (shown in FIG. 8), wherein the process of decoding each variable-length code is conditioned by the result of decoding the preceding variable-length code (emphasis added).

Cocke, col. 3, lines 16 to 25.

Cocke describes Figure 10 as:

> FIG. 10 is **a general block diagram of the independent decoder as a whole, indicating its various components.**

Cocke, col. 3, lines 43 to 45. Thus, it is clear that Cocke's descriptions of the cited figures teach "showing the manner in which the primary and secondary processors P and S are employed to decode individual variable-length codes independently of one another" and "a general block diagram of the independent decoder as a whole, indicating its various components".

More particularly, the cited portions of Cocke are silent as to any teaching, suggestion, or disclosure of <u>a unique variable bit length binary representation of the absolute value of said integer data</u>, as recited in Claim 2. The cited figures

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

and respective descriptions of Cocke do not indicate whether code and values are absolute, positive, non-negative, or negative. In fact, the cited figures and descriptions are silent as to whether the code is integer data, character data, or other. Thus, the Office Action failed to show that Cocke teaches, suggests, or discloses the invention as recited in Claim 2. This alone is sufficient for the withdrawal of the anticipation rejection of Claim 2.

In addition, and with respect to the cited Figure 4, Cocke expressly teaches:

> **GENERAL DESCRIPTION (FIGS. 1-9)**
>
> For an adequate understanding of the invention, some preliminary knowledge of variable-length coding is essential. **One of the well-known types of variable-length coding is Huffman coding, a very elementary example of which is shown in FIG. 1. With some modification (which will be explained hereinafter), this is the type of coding employed in the present illustrative system. In this example, it is assumed that the entire data base under consideration contains only four different characters, A, B, C and D,** which are arranged in various ways for representing information that is to be transmitted or stored. It is further assumed that in this data base the character A is employed most frequently, with exactly one-half of all the characters in the data base being A's. Next in order of frequency (in this particular example) is the character B, with one-quarter of the characters in the assumed data base being B's. The remaining characters are C's and D's, each of which is assumed to constitute one-eighth of all the characters used in the data base.
>
> Since there are only four different characters, any of these characters may be represented in ordinary fixed-length code notation by a distinctive two-bit code word. Thus, by way of example, A may be represented by the bit pair 00, B by 01, C by 10 and D by 11 in the assumed two-bit fixed-length code notation.

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

> In the description which follows, the fixed-length codes often will be referred to as the "ID" or "identity" codes; that is to say, they are identical with codes that are used in a conventional data processing system.
>
> **If it is desired to achieve data compaction, then the characters A, B, C and D, rather than being represented by fixed-length codes which invariably consist of two bits each, are represented instead by codes having variable-length bit strings containing from one to three bits each.** The lengths of such bit strings are inversely related to the frequency with which the respective codes are used. **By properly assigning these variable-length (VL) codes to the respective characters, the average length of the code words utilized to represent characters drawn from this data base may be made less than two bits, thereby saving tansmission time and storage space in handling of such data** prior to its utilization by a conventional data processor (emphasis added).

Cocke, col. 3, lines 62 to 67; col. 4, lines 1 to 42. Cocke further explains the modification to the Huffman method as:

> In Huffman coding, no code may form the beginning of a longer code. However, two different codes of the same length may have a prefix portion in common. In FIG. 1, for example, the three bit codes 110 and 111, respectively representing the characters C and D, have identical two-bit prefix portions 11. Also, in normal Huffman coding, codes of different lengths may have common prefix portions, provided this prefix portion does not constitute an entire code word. Thus, in FIG. 1, the codes 10 and 110, respectively representing B and C, share the one-bit prefix 1.
>
> In order to practice the invention, **a special constraint is imposed upon the Huffman coding process so that no two codes of different lengths exceeding N bits will share the same N-bit prefix, where N is a fixed number representing, for example, the number of input code bits that can conveniently be processed at one time by the available memory hardware. This**

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

> **constraint will reduce somewhat the degree of compaction that may be attained, but the disadvantage is small in comparison with certain advantages (to be described herein) that are realized by fulfilling this condition. With these constrained variable-length codes, each N-bit prefix is uniquely assigned to a particular code length.**

Cocke, col. 5, lines 26 to 51. Thus, Cocke modifies the Huffman method only as to the special constraint that "…no two codes of different lengths exceeding N bits will share the same N-bit prefix, where N is a fixed number representing, for example, the number of input code bits that can conveniently be processed at one time by the available memory hardware". The above-referenced and cited sections of Cocke are silent as to any modification of the Huffman method pertaining to sign representation. Therefore, in teaching the Huffman method for expressing variable length codes, <u>the quoted sections of Cocke teach the Huffman method of sign representation for variable length codes.</u>

Cocke's teaching of use of the Huffman method for sign representation of variable-length codes is in direct contrast to the teaching of the present invention, which states:

> The prediction residual is computed using modulo 2^{16} and is expressed as a pair of symbols, namely the category and the magnitude. As is known to those of ordinary skill, the first symbol, namely the category, represents the number of bits needed to encode the magnitude. **This symbol is Huffman coded.**
>
> For example, if the prediction residual for X is 68, an additional 7 bits are needed to uniquely identify the value 68. This prediction residual is then mapped into a two-tuple (category 7, 7-bit code for 68). The compressed representation for the prediction residual consists of this Huffman codeword for category 7, followed by the 7-bit representation of the magnitude. **In general, if the value of the residual is non-negative, the code for the magnitude is its direct binary representation.**

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

> **If on the other hand, the residual is negative, the code for the magnitude is the one's complement of its absolute value. This means that the codeword for negative residuals always start with a zero bit.**
>
> **As illustrated, prior art contains several methods for representing the signs of fixed- and variable-width data.** The optimal choice of sign representation depends on the encoding scheme being used and on the nature of the data being encoded. The choice will typically be based on issues like CPU performance and ease of programming. **Thus it is useful to introduce a new method of sign representation which, for both encoding and decoding data, is simple to program and requires minimal CPU usage** (emphasis added).

Applicant's Specification, page 3, paragraphs 2 and 3; page 5, paragraph 3. Thus, the present invention expressly avoids the Huffman method of sign representation.

More particularly, one skilled in the art will recognize that the Huffman method of sign representation contains an indication of the sign of integer data in the following manner: <u>(1) if the value of the residual is non-negative, the code for the magnitude is its direct binary representation</u>; and (<u>2) if on the other hand, the residual is negative, the code for the magnitude is the one's complement of its absolute value</u>, Applicant's Specification, above.

The present invention's method, by contrast, includes <u>appending to said unique variable bit length binary representation a single bit representing the sign of said integer data</u>, as recited in Claim 2. Even more particularly with respect to Claim 2, said <u>single bit is zero for integer data that is less than or equal to zero</u>. Thus, Cocke teaches away from the present invention on multiple levels. Each teaching is sufficient for withdrawal of the anticipation rejection of Claim 2.

Further, the section(s) of Cocke set out above teach the use of <u>variable-length codes to represent characters</u>, as

GUNNISON, McKAY &
HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

illustrated with the use of the characters A, B, C, and D. Representation of characters teaches nothing about providing a unique variable bit length binary representation of the absolute value of said integer data, as recited in Claim 2.

One skilled in the art will recognize that character data is mutually exclusive from integer data. Further, a concept of representing character data as an absolute value has no meaning. Therefore, the Cocke teaching of representation of character data cannot teach representation of an absolute value of integer data. The Office Action failed to show that the cited portions of Cocke teach, suggest, or disclose providing a unique variable bit length binary representation of the absolute value of said integer data. This, too, is sufficient for withdrawal of the anticipation rejection of Claim 2.

The Office Action further indicated that one can set the bit as desired to correspond to negative or positive signs (see col. 2, lines 63-67).

The cited section of Cocke actually teaches:

> -register **are shifted by the length of the code, in response to a length value read out of the decoding table, thereby bringing the next succeeding variable-length code into the leading position for addressing the table** (emphasis added).

Cocke, col. 2, lines 63 to 67.

With respect to the foregoing teaching, one skilled in the art will recognize that bit shifting, i.e., shifting the contents of a register, efficiently provides a result of an arithmetic operation using one register, where the result would otherwise have to be obtained by performing the arithmetic operation itself using effort-intensive multiple loads of multiple registers.

The cited section of Cocke in context specifically teaches a particular bit shifting operation where the contents of the "register are shifted by the length of the code, in response to

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

a length value read out of the decoding table, thereby bringing the next succeeding variable-length code into the leading position for addressing the table".

With respect to the foregoing particular bit shifting operation, Cocke further emphasized:

> The same result could have been achieved by reading from field AB of the primary processor memory 12 a value 10000 that directly represented a base address and then arithmetically adding to it the displacement value of 00001, without shifting the contents of the address register 24. **It is preferred, however, to operate in the manner described above, since this avoids an arithmetic addition** (emphasis added).

Cocke, col. 8, lines 21 to 28. Thus, Cocke explicitly teaches one particular bit shifting operation as a preferred means to efficiently obtain <u>bringing the next succeeding variable-length code into the leading position for addressing the table</u>.

This teaches nothing about <u>appending to said unique variable bit length binary representation a single bit representing the sign of said integer data, wherein said single bit is zero for integer data that is less than or equal to zero</u>, as recited in Claim 2.

Assuming arguendo that bit shifting in general can be equated with an append operation, a bit shift by the <u>length of the code</u> cannot be used to teach appending <u>a single bit</u> representing the sign of said integer data.

In general, for example, Cocke teaches cases where the length of the code exceeds one bit:

> **If an input code has a length not exceeding N bits, (assumed to be four bits in the present example)** the processor P furnishes the final decoded output. **If an input code has a length exceeding four bits,** decoding is performed in two stages (emphasis added).

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

Cocke, col. 6, lines 45 to 50. Thus, with respect to the length of an input code, Cocke teaches "[i]f an input code has a length not exceeding N bits, (assumed to be four bits in the present example)" and "if an input code has a length exceeding four bits, decoding is performed in two stages". The respective lengths of the foregoing examples, i.e., four bits and exceeding four bits, cannot be expressed with a single bit, as required by Claim 2. This alone is sufficient for the withdrawal of the anticipation rejection of Claim 2.

Specifically, Cocke cannot teach a length of zero. A length of zero in the particular Cocke bit shifting operation would represent a code having no length, thus no code. This renders a length of zero meaningless, and prevents Cocke from teaching a single bit of zero for integer data that is less than or equal to zero, as recited in Claim 2. This, too, is sufficient for the withdrawal of the anticipation rejection of Claim 2.

In addition, in the particular bit shifting operation of Cocke, a bit shift of zero would not produce the result taught by Cocke. Stated differently, a bit shift of zero produces no change to the contents of the register, and thus cannot bring the next succeeding variable-length code into the leading position for addressing the table. Thus, one might assume that the length of the code taught by Cocke could never be zero. This, too, is sufficient for withdrawal of the anticipation rejection of Claim 2.

Further, as discussed above, Cocke expressly teaches a particular result of the bit shifting operation; namely, bringing the next succeeding variable-length code into the leading position for addressing the table. This bears no relationship to the result of appending a single bit; i.e., representing the sign of said integer data, as recited in Claim 2. This, too, is sufficient for withdrawal of the anticipation rejection of Claim 2.

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

For at least each of the foregoing reasons, the Office Action failed to show that the cited sections of Cocke teach, suggest, or disclose the invention as recited in Claim 2. Applicant respectfully requests reconsideration and withdrawal of the anticipation rejection of Claim 2.

The Examiner rejected each of Claims 3, 4, 5, 7, 8, 10, 11, 13, 14, and 15 for the same reasons given for Claim 2.

Applicant respectfully traverses the anticipation rejection of each of Claims 3, 4, 5, 7, 8, 10, 11, 13, 14, and 15.

As discussed with respect to Claim 2 and herein incorporated by reference, the Office Action failed to show that the cited figures and sections of Cocke teach, suggest, or disclose the invention as recited in each of Claims 3, 4, 5, 7, 8, 10, 11, 13, 14, and 15. For example, the Office Action failed to show that the cited figures and sections of Cocke teach, suggest, or disclose <u>providing a unique variable bit length binary representation of the absolute value of said integer data</u> and <u>appending to said unique variable bit length binary representation a single bit representing the sign of said integer data</u>. Applicant respectfully requests reconsideration and withdrawal of the anticipation rejection of each of Claims 3, 4, 5, 7, 8, 10, 11, 13, 14, and 15.

Claim 9 depends directly from Claim 8. Consequently, Claim 9 includes all of the features and limitations of Claim 8. Therefore, Applicant respectfully submits that Claim 9 is also allowable over Cocke for at least the reasons discussed above.

Claims 2, 3, 4, 5, 7, 8, 9, 10, 11, 13, 14, and 15 remain in the application. For the foregoing reasons, Applicant respectfully requests allowance of all pending claims.

GUNNISON, McKAY &
HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

If the Examiner has any questions relating to the above, the Examiner is respectfully requested to telephone the undersigned Attorney for Applicant.

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on April 28, 2006.

Attorney for Applicant(s) April 28, 2006
Date of Signature

Respectfully submitted,

Philip J. McKay
Attorney for Applicant(s)
Reg. No. 38,966
Tel.: (831) 655-0880

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888